

MOSAIC 64K Select Newsletter

Issues Number
3 and 4



In order to serve you better during
the holiday mail rush, we've combined
issues #3 and #4 into a deluxe double issue
From the Editors Desk;

Being a relatively new company, last year was the first year Mosaic had a Christmas tree. And you know how first trees can be in a rapidly growing company with little cash to spare for such things.

Anyway, not to let our spirits be dampened by the situation, we decided to use our ingenuity by decorating the tree with the wonderful things that are thrown in boxes in the engineering department and stuck on a shelf in the back of the room to collect dust.

Among the bits of colored wire, unused components, and dull X-Acto knives, we made a wonderful discovery. We finally found a use for boards made by the other guys that we purchased with the intent to learn from others mistakes.

We proudly hung them on our tree where we could enjoy them. These boards are no longer stored in a box on a dusty shelf. Their new home is in the closet with the Christmas tree stand where they at least have some hope for a useful future.



Your friends at Mosaic wish you a
happy holiday season!

Handwritten text at top left, possibly a date or reference number.

Handwritten text at top right, possibly a name or title.

Main body of handwritten text, appearing to be a letter or report, spanning across the middle of the page.

Second section of handwritten text, possibly a continuation or a separate paragraph.

Third section of handwritten text, continuing the narrative or report.

Small handwritten note or signature in the center of the page.

Handwritten text at the bottom of the page, possibly a closing or signature.

400 ONLY HARDWARE FIX

We have been getting a number of inquiries about the 400 ONLY RAM SELECT, as to why some software won't boot. Never fear the fix is here.

The problem comes from the software logic that checks to see if there is adequate memory available. Most new software is now written with modified logic that eliminates this problem. But there is still a lot of older software around and will continue to be so.

This fix will give the ability to the user to deselect the bank memory areas on the board. The user will have the option to switch between a 48K or 64K machine. It should be noted that the fix requires many technical skills, including soldering, identifying printed circuit traces and I.C.s. The fix is NOT intended for the non-technical user. So, please don't attempt it if you have even the slightest doubt of your skills.

First, you must disassemble your 400 EXACTLY as you did when you installed the board, being careful not to damage the connector between the mother board and power supply. Next remove the Ram Select board, carefully, removing the 16 pin dip jumper cable from the front. Once this is accomplished and the board is free we can get to the fun part.

You will need a low power soldering iron (25 watts or so), a single pole double throw toggle switch that can be mounted through the 400's case (so you can have easy access to it), exacto knife or equivalent (to cut printed circuit traces, OH NO!), approx. two feet of stranded 22-24 awg wire, one 4.7K ohm 1/4W resistor and a little patience.

1. Locate pin 11 of U17, cut the trace that runs between pins 11 and 12. Make sure you have located the right trace.
2. Solder the 4.7K ohm resistor from pin 11 of U17 to pin 14 of U17. Since pin 14 is Vcc this is a pullup resistor.
3. Using your wire and switch arrangement, solder one side of the switch to pin 11 of U17 and the other to pin 11 of U18.

Once you have completed the above steps you may place the switch anywhere in the 400's case that is convenient. Make sure to note which way the switch is open and closed. The banks will be disabled whenever the switch is open and enabled when closed. When the banks are disabled the machine will only recognize 48K just as any full configured ATARI 800. This fix has worked well in the past for us. We're sure it will cure any compatibility problems you may have had. The same fix is available for the 400-800 board with much less strain. There are strapping posts located in various locations on the board. The one we are concerned with is on the far left of the board (with the card-edge connector towards you) called B. Normally this strap is set to the zero position, meaning board zero. If you move the strap to the one position you can have the best of both worlds. With the strap in the one position, on power up the computer will only recognize 48k but the banks are still available to you. Normally the the first bank is selected by poking location 65472 decimal with any value. When the Bstrap is set at one your first bank will now be selected by poking 65472+16 or 65488. The software boot fix that comes with the 400-800 manual deselects all banks rendering them useless.

M =

If there is one thing that computer industry analysts agree on, it's this, that the microcomputer revolution is just beginning. Several technologies have been developed to make the personal computer more personal. But even with the development of video displays and advanced operating systems such as in the Atari, personal computers are still not very friendly.

Consider what must be done to load and RUN a BASIC program.

```
LOAD Dn:filename  
RUN
```

These are the simplest of commands. They ignore whether or not the proper disk has been inserted and whether or not the DOS directory has been consulted and even whether DOS was loaded in the first place.

All of today's computer technologies are oriented toward the single application environment. This means all programs are loaded and used one at a time. In other words, things aren't nearly as nasty as they can get.

Consider what must be done to collect information from your database program while your in the middle of a letter on your word processor:

```
SAVE, SWAP, LOAD  
SAVE, SWAP, LOAD
```

I may have forgotten a few steps, but I think you get the idea.

The day that computers eliminate as much work as they create is the day that the computer revolution will really begin.

Q: So what does all of this have to do with the 64K Select?

A: The Mosaic Memory Manager!

Today's computer users are fortunate. Before the days of powerful operating systems, the programmer was responsible for selecting proper I/O addresses and proper timing of the mechanical apparatus in order to send data to a printer. Now all of those tasks are performed by a part of the operating system called the printer handler. Most all devices are now provided with some sort of handler. Most, but not all!

Memory is a device, but even today the programmer is still responsible for its management. Memory management is difficult for single application environments.

But the day is soon coming when multiple programs will reside in system memory simultaneously. Computer users should then be able to select and use a program as simply as selecting an option from a menu.

As multiple programs are used simultaneously, the problems of proper memory allocation will grow exponentially. In fact, memory allocation is one of the main obstacles in developing the multi-programming environment.

Mosaic has developed memory allocation specifications for the Mosaic Memory Manager. It encompasses both a user interface and the mechanics of physical memory allocation. What follows is a description of the design goals and allocation structure.

The basic design goal of the memory handler is to "allow the programmer to only be concerned with the procedures that have a direct bearing on the task at hand." The memory handler will perform all chores of memory allocation, data storage, data access, movement and transfer.

In developing the allocation structure, many requirements were taken into consideration:

- A. The memory should allow storage and access to all types of files.
 - 1. Bulk storage similar to a disk drive; these files would require movement of the data from the extended memory to a user specified data buffer.
 - a. Serial ASCII files.
 - b. Random Access data files.
 - 2. Direct access storage similar to normal RAM; these files will be accessed directly by the software.
 - a. Direct access data.
 - b. Directly executable code.
- B. The file structure should allow storage of named files.
- C. Memory should be allocated by sector to provide maximum flexibility.
- D. The allocation structure must not be imbedded with sector link information due to direct access files.
- E. The allocation structure must allow files to be appended thus some type of sector link technique is needed.
- F. Sector sizes should be 256 bytes for maximum flexibility and minimum data manipulation overhead.
- G. 1K sector sizes should also be tracked to further reduce data manipulation overhead and allow compatibility with future memory systems.

From the above stated design goals, the following allocation structure was developed.

A directory table is needed to hold file names and to easily find those files. Each directory entry will require 16 bytes in order to track the following information.

Location,	Byte, # of bytes,	Function
NAM	0-7	8 ASCII name (all bytes 0 if not used)

EXT	8,9	2	ASCII file extender
LOC	10,11	2	Location in extended memory
		1	Bank # (0-63) 64 banks possible
		1	Sector # (0-15) 16 sectors per bank
LEN	12,13	2	File length (0-64K) consecutive bytes
SLT	14,15	2	Location of sector link block (\$C400-\$C5FF, 0 if not used)

The filename can be any valid DOS file name. Note that only two characters can be used for the file extender rather than three. The location of the file can easily be found using the following technique:

```

LDA  #$C0      ;Calculate high byte of file address.
ORA  LOC+1
STA  TEMP+1    ;Store for indirect addressing.
LDA  #0        ;Lo byte = 0.
STA  TEMP
LDX  LOC       ;Select proper memory bank.
STA  BNKBAS,X

```

The length of the file is measured in consecutive bytes. This is possible since no sector link data is stored in the actual sector. However, any program loop that pages through the data must take into consideration the 4K bank boundaries. The length only applies to consecutive bytes. The length of additional non-consecutive bytes are held in the sector link table.

Non-consecutive sectors are created when files are created, then deleted, and a new, larger file is created that fills the holes left by smaller deleted files.

The SLT bytes of the directory hold pointer information to an associated Sector Link Table. They are only necessary when a file cannot be placed in consecutive sectors. If unused, they will hold a value of 0.

Since the Select memory system was designed to allow up to 256K of extended memory and each directory entry requires 16 bytes, it was determined that 1K is needed to hold the directory table. This will allow 64 file names with an average file size of 4K. The directory is the first 1K of bank 0.

The Sector Link Table will require 16 bytes per entry. Each entry is called a block.

Location,	Byte,	# of bytes,	Function
TST	0	1	(\$80 if used, 0 if not used)
ENT	1	1	entries (1-3) up to 3 sector links per block
NSL	2,3	2	location of next sector link block (0 if not used)
SL1	4-7	4	sector link info
		2	location (bank, sector)
		2	length (0-64K)
SL2	8-11	4	sector link info
SL3	12-15	4	sector link info

The sector link table requires 512 bytes from \$C400 to \$C5FF of Bank 0. It allows up to 32 sector link blocks of 16 bytes each. In addition to allowing directly executable code to occupy multiple sectors, the sector link table has another important advantage over imbedded link techniques. True random access files are now possible since sector link information can be found without having to page through the entire data file.

A sector bit map table is needed to track which sectors are available for use. Mosaic has developed an ingenious technique for tracking this information. One byte is required for each 1K of extended memory. The right nibble of each byte will track sector use, since there are 4-256 byte sectors per 1K of RAM. The most significant bit of each byte will be used to signal when all 4 sectors are in use. Through this technique an empty sector or a proper number of consecutive empty sectors can be found very quickly.

When searching for an empty sector, the following technique can be used.

LDA from bit MAP

BMI means all 4 sectors are used

BEQ means all 4 sectors are not used

If necessary, the traditional technique of MASK and COMPARE can then be used to determine which sectors are unused.

In addition to speed, this technique has another advantage; software developed to use it can easily be upgraded to use hard disks and future memory systems where 1K sectors will be common. This sector map technique requires less than 0.1% memory overhead.

The good news: Novice users do not need to understand any of what was just described in order to take advantage of the memory handler. The software that takes advantage of the memory handler will be easier to use than those that use DOS.

The bad news: The memory handler does not exist, yet.

The next issue of the newsletter will describe the command structure of the memory handler. In the meantime, we are requesting input from Select owners. What are your ideas for what a memory handler should do? How do you think it should work?

If you are a machine language programmer, we would like to hear from you. We are looking for someone who is intimately familiar with the Atari OS. Send a letter or give us a call.

1-800-2-ADD-RAM
(800-223-3726)

SUPER-DUPER

By RaN

For Atari 800 computers
With MOSAIC RAM SELECT

The SUPER-DUPER program was written so that a Mosaic RAM Select user may make backup copies of valuable application programs. It was not intended for use by pirates.

Super Duper was designed to take full advantage of the extra bank select areas provided by the RAM Select. It does this by utilizing the extra memory as a temporary storage area between the source and destination disks. It was also meant to be as user friendly as possible by requiring very little user interaction. This program requires 128K of memory. The price for this software package is \$24.95

Write or call 800-2-ADD-RAM to order.

GET INVOLVED

The Select Club was set up for you, the 64K Select owner. We value your comments and suggestions for improvement. We want to print articles on subjects that interest you. Or write an article about a use you've found for the Select board. We'll pay \$50 per page for articles we print. In the near future we will begin a question and answer column also. We want to hear from you. Send questions, comments, and articles to:

The Select Club Editor
C/O Mosaic Electronics
P. O. Box 708
Oregon City, OR 97045

MOZAIC [®] ELECTRONICS
P.O. Box 708
Oregon City, Oregon 97045
(503) 653-7885



BULK RATE
U.S. POSTAGE
PAID
Permit No. 67
Oregon City, OR

